Object Pascal grammar

Topic Groups See Also

```
Goal -> (Program | Package | Library | Unit)
Program -> [PROGRAM Ident ['(' IdentList ')'] ';']
           ProgramBlock '.'
Unit -> UNIT Ident [PortabilityDirective] ';'
        InterfaceSection
        ImplementationSection
        InitSection '.'
Package -> PACKAGE Ident ';'
           [RequiresClause]
           [ContainsClause]
           END '.'
Library -> LIBRARY Ident ';'
           ProgramBlock '.'
ProgramBlock -> [UsesClause]
                Block
UsesClause -> USES IdentList ';'
PortabilityDirective -> platform
                     -> deprecated
                     -> library
InterfaceSection -> INTERFACE
                    [UsesClause]
                    [InterfaceDecl]...
InterfaceDecl -> ConstSection
              -> TypeSection
              -> VarSection
              -> ExportedHeading
ExportedHeading -> ProcedureHeading ';' [Directive]
                -> FunctionHeading ';' [Directive]
ImplementationSection -> IMPLEMENTATION
                         [UsesClause]
                         [DeclSection]...
                         [ExportsStmt]...
Block -> [DeclSection]
         [ExportsStmt]...
         CompoundStmt
         [ExportsStmt]...
ExportsStmt -> EXPORTS ExportsItem [, ExportsItem]...
ExportsItem -> Ident [NAME|INDEX "'" ConstExpr "'"]
                     [INDEX|NAME "'" ConstExpr "'"]
DeclSection -> LabelDeclSection
            -> ConstSection
            -> TypeSection
            -> VarSection
            -> ProcedureDeclSection
LabelDeclSection -> LABEL LabelId
ConstSection -> CONST (ConstantDecl ';')...
ConstantDecl -> Ident '=' ConstExpr [PortabilityDirective]
             -> Ident ':' TypeId '=' TypedConstant [PortabilityDirective]
TypeSection -> TYPE (TypeDecl ';')
TypeDecl -> Ident '=' [TYPE] Type [PortabilityDirective]
         -> Ident '=' [TYPE] RestrictedType [PortabilityDirective]
TypedConstant -> (ConstExpr | ArrayConstant | RecordConstant)
ArrayConstant -> '(' TypedConstant ',' ')'
RecordConstant -> '(' RecordFieldConstant ';'... ')'
RecordFieldConstant -> Ident ':' TypedConstant
Type -> TypeId
     -> SimpleType
     -> StrucType
     -> PointerType
     -> StringType
     -> ProcedureType
     -> VariantType
     -> ClassRefType
RestrictedType -> ObjectType
               -> ClassType
               -> InterfaceType
```

```
ClassRefType -> CLASS OF TypeId
SimpleType -> (OrdinalType | RealType)
RealType -> REAL48
         -> REAL
         -> SINGLE
         -> DOUBLE
         -> EXTENDED
         -> CURRENCY
         -> COMP
OrdinalType -> (SubrangeType | EnumeratedType | OrdIdent)
OrdIdent -> SHORTINT
         -> SMALLINT
         -> INTEGER
         -> BYTE
         -> LONGINT
         -> INT64
         -> WORD
         -> BOOLEAN
         -> CHAR
         -> WIDECHAR
         -> LONGWORD
         -> PCHAR
VariantType -> VARIANT
            -> OLEVARIANT
SubrangeType -> ConstExpr '..' ConstExpr
EnumeratedType -> '(' EnumeratedTypeElement ','... ')'
EnumeratedTypeElement -> Ident [ '=' ConstExpr ]
StringType -> STRING
           -> ANSISTRING
           -> WIDESTRING
           -> STRING '[' ConstExpr ']'
StrucType -> [PACKED] (ArrayType | SetType | FileType | RecType [PACKED])
ArrayType -> ARRAY ['[' OrdinalType ','... ']'] OF Type [PortabilityDirective]
RecType -> RECORD [FieldList] END [PortabilityDirective]
FieldList -> FieldDecl ';'... [VariantSection] [';']
FieldDecl -> IdentList ':' Type [PortabilityDirective]
VariantSection -> CASE [Ident ':'] TypeId OF RecVariant ';'...
RecVariant -> ConstExpr ','... ':' '(' [FieldList] ')'
SetType -> SET OF OrdinalType [PortabilityDirective]
FileType -> FILE OF TypeId [PortabilityDirective]
PointerType -> '^' TypeId [PortabilityDirective]
ProcedureType -> (ProcedureHeading | FunctionHeading) [OF OBJECT]
VarSection -> VAR (VarDecl ';')...
VarDecl  On Windows -> IdentList ':' Type [(ABSOLUTE (Ident | ConstExpr)) | '='
 ConstExpr]     [PortabilityDirective]  On Linux   -> IdentList ':' Type
 [ABSOLUTE (Ident) | '=' ConstExpr] [PortabilityDirective]
Expression -> SimpleExpression [RelOp SimpleExpression]...
SimpleExpression -> ['+' | '-'] Term [AddOp Term]...
Term -> Factor [MulOp Factor]...
Factor -> Designator ['(' ExprList ')']
       -> '@' Designator
       -> Number
       -> String
       -> NIL
       -> '(' Expression ')'
       -> NOT Factor
       -> SetConstructor
       -> TypeId '(' Expression ')'
RelOp -> '>'
      -> '<'
      -> '<='
      -> '>='
      -> '<>'
      -> IN
      -> IS
      -> AS
AddOp -> '+'
      -> '-'
      -> OR
      -> XOR
```

```
MulOp -> '*'
      -> '/'
      -> DIV
      -> MOD
      -> AND
      -> SHL
      -> SHR
Designator -> QualId ['.' Ident | '[' ExprList ']' | '^']...
SetConstructor -> '[' [SetElement ','...] ']'
SetElement -> Expression ['..' Expression]
ExprList -> Expression ','...
Statement -> [LabelId ':'] [SimpleStatement | StructStmt]
StmtList -> Statement ';'
SimpleStatement -> Designator ['(' [ExprList] ')']
                -> Designator ':=' Expression
                -> INHERITED
                -> GOTO LabelId
StructStmt -> CompoundStmt
           -> ConditionalStmt
           -> LoopStmt
           -> WithStmt
           -> TryExceptStmt
           -> TryFinallyStmt
           -> RaiseStmt
           -> AssemblerStmt
CompoundStmt -> BEGIN StmtList END
ConditionalStmt -> IfStmt
                -> CaseStmt
IfStmt -> IF Expression THEN Statement [ELSE Statement]
CaseStmt -> CASE Expression OF CaseSelector ';'... [ELSE StmtList] [';'] END
CaseSelector -> CaseLabel ','... ':' Statement
CaseLabel -> ConstExpr ['..' ConstExpr]
LoopStmt -> RepeatStmt
         -> WhileStmt
         -> ForStmt
RepeatStmt -> REPEAT Statement UNTIL Expression
WhileStmt -> WHILE Expression DO Statement
ForStmt -> FOR QualId ':=' Expression (TO | DOWNTO) Expression DO Statement
WithStmt -> WITH IdentList DO Statement
TryExceptStmt -> TRY
                   Statement...
                 EXCEPT
                   ExceptionBlock
                 END
ExceptionBlock -> [ON [Ident ':'] TypeID DO Statement]...
                  [ELSE Statement...]
TryFinallyStmt -> TRY
                   Statement
                  FINALLY
                   Statement
                  END
RaiseStmt -> RAISE [object] [AT address]
AssemblerStatement -> ASM
                   -> <assemblylanguage>
                   -> END
ProcedureDeclSection -> ProcedureDecl
                     -> FunctionDecl
ProcedureDecl -> ProcedureHeading ';' [Directive] [PortabilityDirective]
                 Block ';'
FunctionDecl -> FunctionHeading ';' [Directive] [PortabilityDirective]
                Block ';'
FunctionHeading -> FUNCTION Ident [FormalParameters] ':' (SimpleType | STRING)
ProcedureHeading -> PROCEDURE Ident [FormalParameters]
FormalParameters -> '(' [FormalParm ';'...] ')'
FormalParm -> [VAR | CONST | OUT] Parameter
Parameter -> IdentList [':' ([ARRAY OF] SimpleType | STRING | FILE)]
          -> Ident ':' SimpleType '=' ConstExpr
Directive -> CDECL
          -> REGISTER
          -> DYNAMIC
```

```
            -> VIRTUAL
            -> EXPORT
            -> EXTERNAL
            -> NEAR
            -> FAR
            -> FORWARD
            -> MESSAGE ConstExpr
            -> OVERRIDE
            -> OVERLOAD
            -> PASCAL
            -> REINTRODUCE
            -> SAFECALL
            -> STDCALL
            -> VARARGS
            -> LOCAL
            -> ABSTRACT
  ObjectType -> OBJECT [ObjHeritage] [ObjFieldList] [MethodList] END
  ObjHeritage -> '(' QualId ')'
  MethodList -> (MethodHeading [';' VIRTUAL]) ';'...
  MethodHeading -> ProcedureHeading
                -> FunctionHeading
                -> ConstructorHeading
                -> DestructorHeading
  ConstructorHeading -> CONSTRUCTOR Ident [FormalParameters]
  DestructorHeading -> DESTRUCTOR Ident [FormalParameters]
  ObjFieldList -> (IdentList ':' Type) ';'
  InitSection -> INITIALIZATION StmtList [FINALIZATION StmtList] END
              -> BEGIN StmtList END
              -> END
  ClassType -> CLASS [ClassHeritage]
               [ClassVisibility]
               [ClassFieldList]
               [ClassMethodList]
               [ClassPropertyList]
               END
  ClassHeritage -> '(' IdentList ')'
  ClassVisibility -> [PUBLIC | PROTECTED | PRIVATE | PUBLISHED]
  ClassFieldList -> (ClassVisibility ObjFieldList) ';'...
  ClassMethodList -> (ClassVisibility MethodList) ';'...
  ClassPropertyList -> (ClassVisibility PropertyList ';')...
  PropertyList -> PROPERTY Ident [PropertyInterface] [PropertySpecifiers]
    [PortabilityDirective]
  PropertyInterface -> [PropertyParameterList] ':' Ident
  PropertyParameterList -> '[' (IdentList ':' TypeId) ';'... ']'
  PropertySpecifiers -> [INDEX ConstExpr]
                        [READ Ident]
                        [WRITE Ident]
                        [STORED (Ident | Constant)]
                        [(DEFAULT ConstExpr) | NODEFAULT]
                        [IMPLEMENTS TypeId]
  InterfaceType -> INTERFACE [InterfaceHeritage]
                   [ClassMethodList]
                   [ClassPropertyList]                ...
                   END
  InterfaceHeritage -> '(' IdentList ')'
  RequiresClause -> REQUIRES IdentList... ';'
  ContainsClause -> CONTAINS IdentList... ';'
  IdentList -> Ident ','...
  QualId -> [UnitId '.'] Ident
  TypeId -> [UnitId '.'] <type-identifier>
  Ident -> <identifier>
  ConstExpr -> <constant-expression>
  UnitId -> <unit-identifier>
  LabelId -> <label-identifier>
  Number -> <number>
String -> <string>
```